Mathematics for Intelligent Systems

BASIS PURSUIT

Rohith. G

TABLE OF CONTENTS

Acknowledgement

The success and final outcome of this project required a lot of guidance and assistance from many people and we extremely fortunate to have got this along the completion of our project work. Whatever we have done is only due to such guidance and assistance and we would not forget to thank them.

First and foremost, I would like to thank the faculties in charge Dr. Soman K. P for giving us an opportunity to do this project work and providing us all support and guidance which made us complete the project on time. who was available at every working hour, ready to solve our doubts and improve our project to yield the best possible result.

ABSTRACT

The alternating direction method of multipliers (ADMM) algorithm is applied to the compressed sensing theory to realize the sparse optimization of vibration signal. Solving the basis pursuit problem for minimizing the L1 norm minimization under the equality constraints, the sparse matrix obtained by the ADMM algorithm can be reconstructed by inverse sparse orthogonal matrix inversion. This paper analyzes common sparse orthogonal basis on the reconstruction results, that is, discrete Fourier orthogonal basis, discrete cosine orthogonal basis, and discrete wavelet orthogonal basis. In particular, we will show that, from the point of view of central tendency, the discrete cosine orthogonal basis is more suitable, for instance, at the vibration signal data because its error is close to zero. Moreover, using the discrete wavelet transform in signal reconstruction there still are some outliers but the error is unstable. We also use the time complex degree and validity, for the analysis of the advantages and disadvantages of the ADMM algorithm applied to sparse signal optimization. The advantage of this method is that these abnormal values are limited in the control range

INTRODUCTION

The alternating direction method of multipliers is a powerful algorithm for solving structured convex optimization problems. The generalization of ADMM's usage is in solving convex optimization problems where the data can be arbitrarily large. While the admm method was introduced for optimization in the 1970's, its origins can be traced back to techniques for solving elliptic and parabolic partial difference equations developed in the 1950's . Admm enjoys the strong convergence properties of the method of multipliers and the decomposability property of dual ascent, and is particularly useful for solving optimization problems that are too large to be handled by generic optimization solvers. The method has found a large number of applications in diverse areas such as compressed sensing, regularized estimation , image processing , machine learning , and resource allocation in wireless networks. This broad range of applications has triggered a strong recent interest in developing a better understanding of the theoretical properties of admm. The generalization of admm's usage is in solving convex optimization problems where the data can be arbitrarily large.

Generally, such issues are solved by using parallel versions of algorithms to distribute the work- load across multiple processors, thus speeding up the optimization. But our traditional optimization algorithms are not suitable for parallel computing, so we must use a method that is. Such a method would have to decentralize the optimization; one good way to do this is to use the alternating direction method of multipliers (admm). This convex optimization algorithm is robust and splits the problem into smaller pieces that can be optimized in parallel.

We will first give some background on admm, then describe how it works, how it is used to solve problems in practice, and how it was implemented. Then, we discuss common problems that admm is used to solve and how they were implemented as general solvers in the admm library. Finally, we discuss our results on adaptive step-size selection and draw conclusions on the potential for this to be used in practice. We end by discussing potential future work on this solver library and adaptive step-sizes.

After briefly surveying the theory and history of the algorithm, we discuss applications to a wide variety of statistical and machine learning problems of recent interest, including the lasso, basis pursuit, covariance and many others.

Lagrangian multipliers:

Here we want to find a minimum or maximum value of function in the given region.

Basic equation for this funcWe have to find $x_1, x_2, ..., x_n$ which satisfies the above inequalities

Lagrangian equation →

$$L(x, \lambda_1 > 0, \lambda_2 > 0, .., \lambda_n > 0) = f(x) - \lambda_1(g_1(x)) - \lambda_2(g_2(x)) - ... - \lambda_n(g_n(x))$$

For this equation optimal points are needed points

To find it the values we will solve the below differential equations

$$\frac{\delta f(x_1, x_2, .., x_N)}{\delta x_1} - \lambda_1 \left(\frac{\delta g(x_1, x_2, .., x_N)}{\delta x_1} \right) = 0$$

$$\frac{\delta f(x_1, x_2, .., x_N)}{\delta x_2} - \lambda_2 \left(\frac{\delta g(x_1, x_2, .., x_N)}{\delta x_2} \right) = 0$$

.

.

$$\frac{\delta f(x_1, x_2, .., x_N)}{\delta x_N} - \lambda_N \left(\frac{\delta g(x_1, x_2, .., x_N)}{\delta x_N} \right) = 0$$

Solution:

$$\nabla_{x,y,\lambda}\mathcal{L}(x,y,\lambda)=0.$$

$$\nabla f(x^*)-\lambda\nabla g(x^*)=0\ vector-------(1)$$

$$g(x^*)\geq 0-------(2)$$

$$\lambda g(x^*)=0-------(3)$$

Where,

$$\nabla_{x,y}f=\left(\frac{\partial f}{\partial x},\frac{\partial f}{\partial y}\right),\qquad \nabla_{x,y}g=\left(\frac{\partial g}{\partial x},\frac{\partial g}{\partial y}\right)$$

3rd condition is complimentary condition

At boundaries $g(x^*)=0$ so $\lambda>0$

Inside the boundary $g(x^*)>0$ so $\lambda=0$

AUGMENTED LAGRANGIAN AND THE METHOD OF MULTIPLIERS:

Augmented lagrangian method was developed to bring robust to the dual ascent method. The augmented lagrangian is denoted as $L_\rho(x, y)$. The augmented lagrangian for minimizing the function $f(x)$ subjected to $Ax = b$ is

$$L_\rho(x, y) = f(x) + \lambda(Ax - b) + (\rho/2) \| Ax - b \|_2^2$$

where ρ>0, it is called penalty parameter and $(\rho/2) \| Ax - b \|_2^2$ is called as penalty term after modifying this problem is clearly equivalent to the original problem since for any feasible x the term added to the objective is zero.

Now we have to minimize the x value. for that we can apply dual ascent to our modified problem.

$$x^{k+1} = \arg_x \min L_\rho(x, z^k, \lambda^k)$$
$$z^{k+1} = \arg_z \min L_\rho(x^{k+1}, z\ , \lambda^k)$$
$$\lambda^{k+1} = \lambda^k + \rho(Ax^{k+1} + Bz^{k+1} - c)$$

At first, we have to perform x-minimization and then z-minimization step. Thus, they update alternatively.

ADMM write-up

$$\min_x f(x) + g(x)$$

$x, u, z \in R^n$,

$\lambda \in \mathrm{R}^+$ a hyper parameter

$$x^{k+1} := prox_{\lambda f}\left(z^k - u^k\right)$$
$$z^{k+1} := prox_{\lambda g}\left(x^{k+1} + u^k\right)$$
$$u^{k+1} := u^k + x^{k+1} - z^{k+1}$$

z-surrogate variable to x-variable to be minimised

u-lagrangian multipliers

If f(x) and g(z) are differentiable, Then

$$x^{k+1} = \underbrace{\arg\min}_{x} f(x) + \frac{1}{2\lambda}\left\|x - \left(z^k - u^k\right)\right\|_2^2$$

$$\Rightarrow x^{k+1} \text{ is the solution of } \nabla f(x) + \frac{1}{\lambda}\left(x - \left(z^k - u^k\right)\right) = 0 \text{ vector}$$

$$z^{k+1} = \underbrace{\arg\min}_{z} g(z) + \frac{1}{2\lambda}\left\|z - \left(x^{k+1} + u^k\right)\right\|_2^2$$

$$\Rightarrow z^{k+1} \text{ is the solution of } \nabla f(z) + \frac{1}{\lambda}\left(z - \left(x^{k+1} + u^k\right)\right) = 0 \text{ vector}$$

$$u^{k+1} = u^k + x^{k+1} - z^{k+1}$$

x,z (Hight dimension parabola)

Increase number of variables from one set to three set , solve one set at a time,

$$\min f(x) + g(z)$$
$$subject\ to \quad x - z = 0$$

$$x - z = \mathbf{0}\ \text{vector} \triangleq \begin{pmatrix} x_1 - z_1 = 0 \\ x_2 - z_2 = 0 \\ \vdots \\ x_n - z_n = 0 \end{pmatrix}$$ Vector form

$$L_\rho(x, y, z) = f(x) + g(z) + y^T(x - z) + \left(\rho/2\right)\|x - z\|_2^2$$

Here,

- y is lagrangian multiplier term

i.e.

$$y_1\left(x_1 - z_1\right) + y_2\left(x_2 - z_2\right) + \ldots + y_n\left(x_n - z_n\right)$$

- $\left(\rho/2\right)$ is augmented langrangian term

Breaking down the problem:

$$x^{k+1} := \arg\min_{x} L_{\rho}\left(x, z^{k}, y^{k}\right)$$

$$z^{k+1} := \arg\min_{z} L_{\rho}\left(x^{k+1}, z, y^{k}\right)$$

$$y^{k+1} := y^{k} + \rho\left(x^{k+1} - z^{k+1}\right)$$

$$L_{\rho}(x,y,z) = f(x) + g(z) + y^{T}(x-z) + \frac{\rho}{2}\|x-z\|_{2}^{2}$$

$$= f(x) + g(z) + \underbrace{y^{T}x}_{\langle y\,,\,x\rangle} - \underbrace{y^{T}z}_{\langle y\,,\,z\rangle} + \frac{\rho}{2}(x-z)^{T}(x-z)$$

The lagrangian function is solved 'one set' of variable *at a time.*
In the $k+1$ *th iteration for computing* x, *we assume* y^{k}, z^{k} *is known .*
x^{k}, y^{k}, z^{k} *are values of* x, y *and* z *obtained in* k^{th} *iteration*

Consider minimization *of Lagrangian function L w.r.t x,*
assuming $y = y^{k}$ *and* $z = z^{k}$ *is known.*

Lagrangian function

$$L_\rho(x) = f(x) + \underbrace{g(z^k)}_{cons\tan t} + \underbrace{x^T y^k}_{\langle y\ ,\ x\rangle} - \underbrace{\left(y^k\right)^T z^k}_{cons\tan t} + \frac{\rho}{2}(x - z^k)^T(x - z^k)$$

Omiting constant terms from objective function, we get

$$L_\rho(x) = f(x) + \underbrace{x^T y^k}_{\langle y\ ,\ x\rangle} + \frac{\rho}{2}(x - z^k)^T(x - z^k)$$

So the update for x can be written as

$$x^{k+1} = \arg\min_x \left(f(x) + \underbrace{\left\langle y^k, x \right\rangle}_{y^T x} + \frac{\rho}{2}\left\| x - z^k \right\|_2^2 \right)$$

You can think that , you obtained x^{k+1} *by solving* $\frac{\partial L_\rho(x)}{\partial x} = 0$ *vector*

We assume x and y vectors are known.

We rewrite Lagrangian as follows:-

$$L_\rho(z) = \underbrace{f(x^{k+1})}_{cons\tan t} + g(z) + \underbrace{\left(y^k\right)^T x^{k+1}}_{Cons\tan t} - \left(y^k\right)^T z + \frac{\rho}{2}(x^{k+1} - z)^T(x^{k+1} - z)$$

Omiting constant terms from objective function, we get

$$L_\rho(z) = g(z) - \underbrace{\left(y^k\right)^T z}_{\langle y\ ,\ z\rangle} + \frac{\rho}{2}(x^{k+1} - z)^T(x^{k+1} - z)$$

$$z^{k+1} = \arg\min_{z} \left(g(z) - \left\langle y^k, z \right\rangle + \frac{\rho}{2} \left\| x^{k+1} - z \right\|_2^2 \right)$$

You can think that , you obtained z^{k+1} *by solving* $\frac{\partial L_\rho(z)}{\partial z} = 0$ *vector*

y vector elements are lagrangian multipliers.

The iterative algorithm proceeds towards (x^*, z^*, y^*) which is a saddle point of L.

This means , w.r.t x, z variables the lagrangian function L is minimum at (x^*, z^*, y^*).

Also, w.r.t y variables the lagrangian function L is maximum at (x^*, z^*, y^*).

So, y vector is updated towards the maximum gradient of L with respect to y

This direction is given by $\frac{\partial L}{\partial y}$

$$L_\rho(y) = f(x^{k+!}) + g(z^{k+1}) + y^T (x^{k+1} - z^{k+1}) + \frac{\rho}{2} \left\| x^{k+1} - z^{k+1} \right\|_2^2$$

Omiting constant terms from objective function, we get $L_\rho(y) = y^T (x^{k+1} - z^{k+1})$

$\frac{\partial L_\rho(y)}{\partial y} = (x^{k+1} - z^{k+1}) \equiv$ *gradient* of L w.r.t y

So to move towards maximum , y is moved along gradient direction from the current position

$$\therefore y^{k+1} = y^k + (step\_size) \times \left(gradient \text{ of L w.r.t y} \right)$$

$$y^{k+1} = y^k + \rho\left(x^{k+1} - z^{k+1} \right) \quad . \text{ We asumed } step\_size \text{ as } \rho$$

We combine 2nd and 3rd term in the first two optimization into a single term,

$$x^{k+1} = \arg\min_x \left(f(x) + \left\langle y^k, x \right\rangle + \frac{\rho}{2} \left\| x - z^k \right\|_2^2 \right)$$

$$z^{k+1} = \arg\min_z \left(g(z) - \left\langle y^k, z \right\rangle + \frac{\rho}{2} \left\| x^{k+1} - z \right\|_2^2 \right)$$

$$y^{k+1} := y^k + \rho\left(x^{k+1} - z^{k+1} \right)$$

On transforming

$$x^{k+1} = \arg\min_x \left(f(x) + \frac{\rho}{2} \left\| x - z^k + \frac{1}{\rho} y^k \right\|_2^2 \right)$$

$$z^{k+1} = \arg\min_z \left(g(z) + \frac{\rho}{2} \left\| x^{k+1} - z + \frac{1}{\rho} y^k \right\|_2^2 \right)$$

$$y^{k+1} = y^k + \rho\left(x^{k+1} - z^{k+1} \right)$$

We verify it by expanding it:

Consider first optimization :

$$Let\ L_\rho(x) = f(x) + \frac{\rho}{2}\left\| x - z^k + \frac{1}{\rho} y^k \right\|_2^2$$

$$= f(x) + \frac{\rho}{2}\left(\left(x - z^k \right) + \frac{1}{\rho} y^k \right)^T \left(\left(x - z^k \right) + \frac{1}{\rho} y^k \right)$$

$$= f(x) + \frac{\rho}{2}\left\{ \left(x - z^k \right)^T \left(x - z^k \right) + 2\left(x - z^k \right)^T \frac{1}{\rho} y^k + \left(\frac{1}{\rho} y^k \right)^T \left(\frac{1}{\rho} y^k \right) \right\}$$

$$= f(x) + \frac{\rho}{2}\left(x - z^k \right)^T \left(x - z^k \right) + x^T y^k + \text{constant}$$

On rearranging

$$L_\rho(x) = f(x) + x^T y^k + \frac{\rho}{2}\left\| x - z^k \right\|_2^2$$

$$So,\ x^{k+1} = \arg\min_x f(x) + \left\langle x, y^k \right\rangle + \frac{\rho}{2}\left\| x - z^k \right\|_2^2$$

Consider second optimization term :

$$Let\ L_\rho(z) = g(z) + \frac{\rho}{2}\left\| x^{k+1} - z + \frac{1}{\rho} y^k \right\|_2^2$$

$$= g(z) + \frac{\rho}{2}\left(x^{k+1} - z + \frac{1}{\rho} y^k \right)^T \left(x^{k+1} - z + \frac{1}{\rho} y^k \right)$$

$$= g(z) + \frac{\rho}{2}\left\{ \left(x^{k+1} - z \right)^T \left(x^{k+1} - z \right) + 2\left(x^{k+1} - z \right)^T \frac{1}{\rho} y^k + \left(\frac{1}{\rho} y^k \right)^T \left(\frac{1}{\rho} y^k \right) \right\}$$

$$= g(z) + \frac{\rho}{2}\left(x^{k+1} - z \right)^T \left(x^{k+1} - z \right) - z^T y^k + \text{constant}$$

On rearranging

$$L_\rho(z) = g(z) - z^T y^k + \frac{\rho}{2}\left\| x^{k+1} - z \right\|_2^2$$

$$So, z^{k+1} = \arg\min_z g(z) - \left\langle z, y^k \right\rangle + \frac{\rho}{2}\left\| x^{k+1} - z \right\|_2^2$$

Now simplifing it,

With

$$u^k = \frac{1}{\rho} y^k, \quad \lambda = \frac{1}{\rho}$$

Hence,

$$x^{k+1} = \arg\min_{x} \left(f(x) + \frac{\rho}{2} \left\| x - z^{k} + \frac{1}{\rho} y^{k} \right\|_2^2 \right)$$

$$z^{k+1} = \arg\min_{z} \left(g(z) + \frac{\rho}{2} \left\| x^{k+1} - z + \frac{1}{\rho} y^{k} \right\|_2^2 \right)$$

$$y^{k+1} = y^{k} + \rho \left(x^{k+1} - z^{k+1} \right)$$

Start with augmented Lagrangian function in the following form,

$$L_{\lambda}(x,u,z) = f(x) + g(z) + \left(\frac{1}{2\lambda} \right) \| x - z + u \|_2^2$$

instead of

$$L_{\rho}(x,y,z) = f(x) + g(z) + y^{T}(x-z) + \left(\rho / 2 \right) \| x - z \|_2^2$$

$$\begin{aligned} &\min && f(x) \\ &subject && to \quad x \in C \end{aligned}$$

This is converted into,

$$\begin{aligned} &\min && f(x) + g(z) \\ &subject && to\, x - z = 0 \end{aligned}$$

Where, Indicator function is

$$g(z) = \begin{Bmatrix} 0 & if\ z \in C \\ \infty & otherwise \end{Bmatrix}$$

BASIC PURSUIT

ADMM is a natural fit for machine learning and statistical problems in particular. The reason is that, unlike dual ascent or the method of multipliers, ADMM explicitly targets problems that split into two distinct parts, f and g, that can then be handled separately. Problems of this form are pervasive in machine learning, because a significant number of learning problems involve minimizing a loss function together with a regularization term or side constraints. In other cases, these side constraints are introduced through problem transformations like putting the problem in consensus form.
This section contains a variety of simple but important problems involving L1 norms. There is widespread current interest in many of these problems across statistics, machine learning, and signal processing, and applying ADMM yields interesting algorithms that are state-of-the-art, or closely related to state-of-the-art methods. We will see that ADMM naturally lets us decouple the nonsmoothed L1 term from the smooth loss term, which is computationally advantageous. We will see the non-distributed versions of these problems for simplicity the problem of distributed model fitting will be treated in the sequel.

The idea of L1 regularization is decades old, and traces back to Huber's work on robust statistics and a paper of Claerbout in geophysics. There is a vast literature, but some important modern papers are those on total variation denoising, soft

thresholding, the lasso, basis pursuit, compressed sensing, and structure learning of sparse graphical models. Because of the now widespread use of models incorporating an L1 penalty, there has also been considerable research on optimization algorithms for such problems. A recent survey by Yang et al. compares and benchmarks a number of representative algorithms, including gradient projection, homotropy methods, iterative shrinkage-thresholding, proximal gradient augmented Lagrangian methods, and interior-point methods. There are other approaches as well, such as Bregman iterative algorithms and iterative thresholding algorithms implementable in a message-passing framework.

BP finds the least l1- norm solution of the underdetermined linear system Ax = b and is used, for example, in compressed sensing for reconstruction.

A good proxy for finding the sparsest solution to an underdetermined system of equations Ax = b is to solve: -

$$\text{minimize } \|x\|_1 \text{ subject to } Ax = b$$

$$ADMM \text{ formulation:}$$

$$\text{minimize } f(x) + g(z) \quad \text{subject to } x - z = \boldsymbol{0} \text{ vector}$$

where,

$$f(x) = \|x\|_1$$

$$g(z) = \begin{cases} 0, & Az = b \\ \infty & \text{otherwise} \end{cases}$$

Suppose we got approximate z =z_{app} in an iteration and $Az_{app} \neq b$

What we need is $A(z_{app} + e) = b$

$$Ae = b - Az_{app} \Rightarrow e = pinv(A) \times \left(b - Az_{app}\right)$$

$$\Rightarrow z^{k+1} = z_{app} + e = z_{app} + pinv(A) \times \left(b - Az_{app}\right)$$

Augmented lagrangian is

$$L(x,z,u) = \|x\|_1 + g(z) + \frac{1}{2\lambda}\|x - z + u\|_2^2$$

$$x^{k+1} = \arg\min_x \|x\|_1 + \frac{1}{2\lambda}\left\|x - z^k + u^k\right\|_2^2$$

$$= S_\lambda\left(z^k - u^k\right)$$

$$\min_x f(x) = \|x\|_1 + \frac{1}{2\lambda}\|x - c\|^2$$

$$sign(x) + \frac{1}{\lambda}(x - c) = 0 \Rightarrow x^* = c - \lambda \times sign(c)$$

$$x^* = S_\lambda(c) = \begin{cases} c - \lambda \times sign(c), & \text{if } \lambda \le |c|, \\ 0, & \text{if } \lambda > |c| \end{cases}$$

$$z^{k+1} = \arg\min_x g(z) + \frac{1}{2\lambda}\left\|x^{k+1} - z + u^k\right\|_2^2$$

$$z^{k+1} = P_{Az=b}\left(x^{k+1} + u^k\right) = \left(x^{k+1} + u^k\right) + pinv(A) \times \left(b - A\left(x^{k+1} + u^k\right)\right)$$

$$u^{k+1} = u^k + \left(x^{k+1} - z^{k+1}\right)$$

Again, the comments on efficient computation by caching a factorization of AAT, subsequent projections are much cheaper than the first one. We can interpret ADMM for basis pursuit as reducing the solution of a least L1 norm problem to solving a sequence of minimum Euclidean norm problems. For a discussion of similar algorithms for related problems in image processing, see. A recent class of algorithms called *Bregman iterative methods* have attracted considerable interest for solving L1 problems like basis pursuit. For basis pursuit and related problems, *Bregman iterative regularization* is equivalent to the method of multipliers, and the *split Bregman method* is equivalent to ADMM.

Basic pursuit using MATLAB:

```
clc;clear all;close all;
n = 30;
m = 10;
A = randn(m,n);
 x = sprandn(n, 1, 0.1*n);
b = A*x;
 xtrue = x;
[X,Z,U] = basis_pursuit(A, b, 1.0, 1.0);
function [x,z,u] = basis_pursuit(A, b, rho, alpha)
    MAX_ITER = 1000;
    [m n] = size(A);
    x = zeros(n,1); z = zeros(n,1); u = zeros(n,1);


    % precompute static variables for x-update (projection on to
Ax=b)
    AAt = A*A';
    P = eye(n) - A' * (AAt \ A);
    q = A' * (AAt \ b);
```

```
    for k = 1:MAX_ITER
        % x-update
        x = P*(z - u) + q;

        % z-update with relaxation
        zold = z;
        x_hat = alpha*x + (1 - alpha)*zold;
        z = shrinkage(x_hat + u, 1/rho);

        u = u + (x_hat - z);
    end
end

function y = shrinkage(a, kappa)
    y = max(0, a-kappa) - max(0, -a-kappa);
end
```

MATLAB OUTPUT:

```
x =

   -0.2890
    0.0674
   -0.0000
    1.4330
    0.0000
    0.8342
    0.0000
    0.0000
    0.0000
    0.0000
   -0.4046
    0.0000
    0.0000
    0.0000
    0.0000
    0.0000
    1.0683
   -0.0000
    0.0000
   -0.1616
    0.0000
    0.0000
   -0.0000
    0.0000
   -0.0000
   -0.8573
   -0.0000
   -0.3402
    0.0742
   -0.0000
```

```
z =

   -0.2890
    0.0674
         0
    1.4330
         0
    0.8342
         0
         0
         0
         0
   -0.4046
         0
         0
         0
         0
         0
    1.0683
         0
         0
   -0.1616
         0
         0
         0
         0
         0
   -0.8573
         0
   -0.3402
    0.0742
         0
```

```
U =

   -1.0000
    1.0000
   -0.1556
    1.0000
    0.5231
    1.0000
    0.0650
    0.1608
   -0.9156
    0.5823
   -1.0000
    0.9714
    0.1099
   -0.2450
    0.2028
   -0.5998
    1.0000
    0.8691
    0.8763
   -1.0000
    0.3713
   -0.1767
   -0.0020
    0.6521
    0.0071
   -1.0000
   -0.4762
   -1.0000
    1.0000
   -0.8886

>>
```

DENOISING USING BASIS PURSUIT

The problem is a convex quadratic problem, so it can be solved by many general solvers, such as interior-point methods. For very large problems, many specialized methods that are faster than interior-point methods have been proposed.

Several popular methods for solving basis pursuit denoising include the in-crowd algorithm (a fast solver for large, sparse problems), homotopy continuation, fixed-point continuation (a special case of the forward–backward algorithm) and spectral projected gradient for L1 minimization (which actually solves LASSO, a related problem).

$$\min_x \left(\frac{1}{2} \|y - Ax\|_2^2 + \lambda \|x\|_1 \right),$$

where λ is a parameter that controls the trade-off between sparsity and reconstruction fidelity, X is an Nx1 solution vector, y is an Mx1 vector of observations, A is an MxN transform matrix and M<N. This is an instance of convex optimization and also of quadratic programming.

$$\min_x \|x\|_1 \text{ subject to } \|y - Ax\|_2^2 \leq \delta,$$

which, for any given λ, is equivalent to the unconstrained formulation for some value of δ. The two problems are quite similar. In practice, the unconstrained formulation, for which most specialized and efficient computational algorithms are developed, is usually preferred.

Either types of basis pursuit denoising solve a regularization problem with a trade-off between having a small residual (making y close to Ax in terms of the squared error) and making x simple in the L1 -norm sense. It can be thought of as a mathematical statement of Occam's razor, finding the simplest possible explanation (i.e. one that yields L1 minization of x) capable of accounting for the observations y.

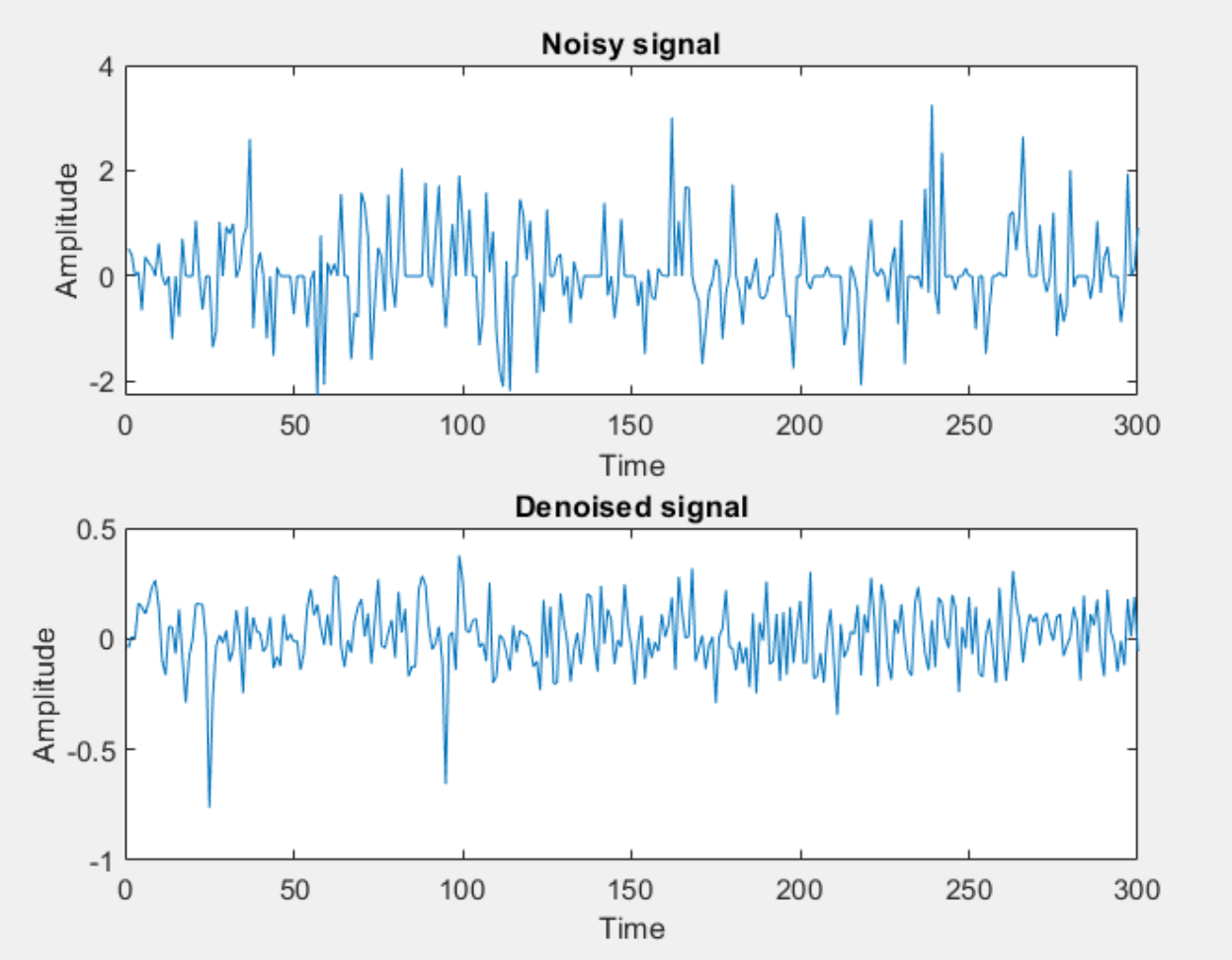
Denoising using Basic pursuit by MATLAB:

```
clc;clear all;close all;
n = 300;
m = 10;
A = randn(m,n);
x = sprandn(n, 1, 0.1*n);
b = A*x;
xtrue = x;
[X,Z,U] = basis_pursuit(A, b, 1.0, 0.003);
subplot(2,1,1)
plot(x)
xlabel('Time');
ylabel('Amplitude')
title("Noisy signal");
subplot(2,1,2)
plot(X)
xlabel('Time');
ylabel('Amplitude')
title("Denoised signal");
```

```
function [x,z,u] = basis_pursuit(A, b, rho, alpha)
% basis_pursuit  Solve basis pursuit via ADMM
MAX_ITER = 1000;
    [m n] = size(A);
    x = zeros(n,1);
    z = zeros(n,1);
    u = zeros(n,1);
% precompute static variables for
% x-update (projection on to Ax=b)
    AAt = A*A';
    P = eye(n) - A' * (AAt \ A);
    q = A' * (AAt \ b);
for k = 1:MAX_ITER
        % x-update
        x = P*(z - u) + q;
        % z-update with relaxation
        zold = z;
        x_hat = alpha*x + (1 - alpha)*zold;
        z = shrinkage(x_hat + u, 1/rho);
        u = u + (x_hat - z);
```

```
    end
end
function y = shrinkage(a, kappa)
    y = max(0, a-kappa) - max(0, -a-kappa);
end
```

MATLAB OUTPUT:



CONCLUSION

In the aspect of error, the mean of ADMM is the minimum and the median is after OMP in view of the central tendency. It indicates the reconstruction error of ADMM is the best and the OMP takes second place. And in view of the dispersion tendency, despite the range of ADMM is not the minimum of three, the variance and the standard deviation are quite small. So, ADMM possesses the good performance in sparse reconstruction compared with primal-dual interior point algorithm and OMP. The insufficient of the algorithm is the present of abnormal value which is not obvious through range. As a result, ADMM sparse optimization algorithm to deal with the problem of sparse reconstruction in compressed sensing has good performance.